Slovenské vzory dělení slov: čas pro změnu?

Petr Sojka

Masarykova univerzita v Brně, Fakulta informatiky*
Botanická 68a, 602 00 Brno
Email: sojka@fi.muni.cz

Abstrakt: Dělení slov neboli algoritmická segmentace velké množiny řetězců nějakého jazyka je problém častější než by se na první pohled zdálo. Pro volně šiřitelné slovenské dělení slov zatím existuje pouze řešení vycházející z definice slabiky ve slovenštině, bez rozsáhlého pokrytí výjimek. Z více než miliónu shromážděných a rozdělených slov se podařilo vygenerovat programem PATGEN nové volně šiřitelné vzory, které se s nepravidelnostmi jazyka vyrovnávají lépe než dosud dostupné řešení. Výsledek je použitelný nejen v distribucích TEXu, ale i v dalších systémech jako například OPENOFFICE.ORG. Použité a diskutované techniky bootstrappingu, stratifikace a generování vzorů jsou použitelné při řešení širokého spektra dalších „segmentačních“ aplikací.

Klíčová slova: dělení slov, segmentace, PATGEN, přebíjející vzory, bootstrapping, stratifikace

1 Motivace

Dělení slov je v jádru všech aplikací pro zpracování textů. Na kvalitě použitého algoritmu dělení slov závisí množství ruční práce při řádkovém zlomu sazby. Stále častější jsou aplikace, kdy kontrola zlomu se neprovádí vůbec: databázové publikování, dávkové zpracování XML dat může sloužit jako příklad. O to větší je poptávka po kvalitním dělení slov. Obvyklé požadavky na algoritmus dělení slov jsou tyto:

rychlost: při optimalizaci zlomu celého odstavce naráz je potřeba najít dělení *všech* slov v odstavci.
přesnost: algoritmus neoznačí chybně švy slov pro rozdělení.
úplnost: algoritmus najde všechna možná dělení slov.
rozšiřitelnost: algoritmus umožní uživatelem specifikované výjimky – například slova cizího jazyka dle pravidel dělení tohoto jazyka.
adaptivita: jelikož se živé jazyky vyvíjejí (nedávná reforma pravopisu v Německu), je potřebné nemít algoritmus „zadrátovaný“ a draze optimalizovaný tak, že při změně jazyka se musí začínat úplně znova.

* Výzkumný záměr CEZ:J07/98:143300003

Petr Olšák (editor): SLT 2004 – sborník semináře o Linuxu a TEXu, str. 67–72, 2004.

parametrizovatelnost: algoritmus umožní jiné chování dle charakteru použití (v úzkých sloupcích například umožní jen dva znaky na novém řádku místo obvykle žádaných tří).

minimální paměťové nároky: aplikace typu zalomení zpráv na displeji mobilního telefonu je třeba navrhovat s minimálními paměťovými, a tedy energetickými nároky.

Základním problémem tedy je vytvořit algoritmus pro zvolený jazyk, který v maximální míře splňuje výše uvedené požadavky. Tento článek popisuje přístup řešení tohoto problému pro slovenštinu a krátce diskutuje výsledek ve formě nových vzorů dělení slov pro slovenštinu.

2 Stávající stav slovenského dělení slov

Uživatelé sázecího systému TEX používají při sazbě výsledky dizertační práce [16] Knuthova studenta Franka Lianga. Liang navrhl na jazyku nezávislý popis dělení slov, který splňuje většinu výše uvedených požadavků. Dále implementoval program PATGEN [17], který umožňuje tento popis generovat ze slovníku již rozdělených slov.

Brzy po rozšíření TEXu do Československa začala být otázka dělení aktuální a vznikly první verze slovenských a českých vzorů dělení [14,15]. Obě byly psány ručně, bez použití programu PATGEN. Ručně psaná pravidla zachycují základní charakteristiky dělení, tedy například slabičný princip, definují co je to slabika. U etymologického přístupu k dělení slov, který je respektován v britské angličtině a standardizován nakladatelstvím Oxford Universitzy Press, však je téměř každé slovo výjimkou a proto je častější generování vzorů ze slovníku. Většina jazyků však oba přístupy kombinuje, ctí se zejména zlom na švech složených slov oproti slabičnému principu. Hranice mezi cítěním složeného slova dle jeho etymologie však může být diskutabilní: máme dělit slabičně *ro-zum* či „etymologicky“ a konzervativně *roz-um*?

Pro detailní vysvětlení principu přebíjejících vzorů odkazujeme na [13, příloha H] a na články [2,22,3]. Zjednodušeně řečeno, vzory specifikují kontextová pravidla, která mezi sebou soutěží o každou mezipísmennou pozici ve slově, a určují, zda na ní dělit či ne. Pravidla specifikují na základě různě širokého kontextu výjimky, výjimky z výjimek,.... Vzájemně se přebíjejí – může existovat několik úrovní priorit vynucení či potlačení dělení slova. Vítězí „nejsilnější“ pravidlo (s nejvyšší prioritou) pro každou pozici ve slově: k pozici se může vyjádřit pravidlo v každé úrovni.

Stávající verze slovenských vzorů dělení existuje ve verzi 2.0 z 24. 4. 1992 (soubor `skhyph.tex`):

```
\patterns{
 % samohl\’asky
a1
\’a1
\"a1
e1
...
 % dvojice spoluhl\’asok
```

```
2b1b
2b1c
2b1\v c
2b1d
2b1\v d
...
 % 6 spoluhl\'asok
3c4v4r4n3g4n
3\v s4k4v4r4k3n
3\v s4k4v4\'r4k3n
 % koncovka -n\'y
k4\v c3n\'y.
k4\v c3n\'eho.
k4\v c3n\'emu.
k4\v c3nom.
 % slovn\'e z\'aklady
5alkohol
auto4rk
auto4rs
5b4lah
5b4ledn
5b4lesk
...
 %koncovky
4b4s4\v t.
8c4h.
8d4z.
8d4\v z.
4c4ht4.
4j4s4\v t.
4lt.
4m4p4r.
...
 %cudzie slov\'a
akci3a2
akv\'ari3u2m
gymn\'azi3um
le2u3k\'emia
t2ri3u2mf
kli3e2nt
}
```

Z komentářů ve vzorech je vidět, jakým způsobem vzory vznikaly. Po rozgenerování vzorů popisujících slabiku jako sekvenci příslušného počtu souhlásek a samohlásek se vzory autorka snažila zachytit slabičné výjimky na začátku a konci slov a při dělení cizích slov. Lze si ale těžko představit, že by se tímto způsobem podařilo zachytit několik miliónů slovních tvarů, které ve slovenštině existují. Na švech předpon a složených slov jsou mnohé výjimky, které jdou proti základnímu slabičnému principu. Těch jsou ale tisíce, či desetitisíce, a jen s enormním úsilím by se daly vypsat všechny. Pro češtinu byly sepsány Hallerem [10], pro slovenštinu však patrně takový soupis neexistuje.

Na archívu CTAN lze nalézt vzory vytvářené jak ručně výše popsaným způsobem, tak automaticky z již rozděleného slovníku slov daného jazyka. Tento postup má z hlediska požadavků vytčených v úvodu článku mnohé výhody oproti ručně vytvořené verzi. Přístupy se také dají kombinovat: k ručně zadané množině základních vzorů se dogenerují vzory pro výjimky. Nebo naopak ex post nalezené výjimky se dají k již vygenerovaným vzorům přidat jako slova – vzory s nejvyšší prioritou (úrovní), tedy *rozum* jako `.r8o8z9u8m.`.

3 Generování vzorů ze slovníku rozdělených slov

Problematice generování vzorů na semináři SLT již byl věnován článek [1], proto zopakujeme jen hlavní principy a laskavého čtenáře odkážeme dále na další články věnované této a příbuzné problematice [11,25,20,21].

Generování probíhá ve fázích, které se nazývají *úrovně* (anglicky *levels*). V lichých úrovních se generují pokrývací vzory, tedy vzory, které dle kontextu znaků vynucují dělení, v sudých úrovních se dělení dle kontextu zakazuje.

Generované vzory se kumulují, a výsledné chování určuje výsledná množina vzorů vygenerovaná ve všech úrovních. U většiny generovaných vzorů pro dělení slov v užívaných jazycích stačí čtyři úrovně, ale pro přehlednost, ale také nedostatek času vzory optimalizovat, je v ručně chystaných vzorech úrovní mnohem více – současné slovenské vzory jich mají například osm.

Technologie přebíjejících vzorů je natolik obecná, že její použití je možné pro většinu *segmentačních problémů*. Jako příklad může sloužit problematika segmentace řetězce thajských znaků na slova – v thajském textu nejsou slova oddělena mezerami [23].

4 Bootstrapping a stratifikace

V rámci bakalářské práce [18] se podařilo shromáždit z různých zdrojů[1] téměř milión slovenských slov. Dnešní výpočetní kapacity umožňují generovat vzory dělení i z takto rozsáhlých slovníků v dobách desítek minut. Časově nejnáročnější operaci – rozdělení slovníku slov pravidly daného jazyka – lze dělat pomocí předchozí verze vzorů a místa dělení slov „pouze“ zkontrolovat. Jelikož však i tato kontrola je časově náročná, lze parametry generování vhodnou heuristikou volit tak, že vygenerovaných vzorů nepokrytých slov je právě tolik, kolik je reálné jich v rozumné době ručně zkontrolovat. To značně urychluje vývoj nových vzorů technikou *bootstrappingu*.

Tabulka 1. Výsledky jedné iterace bootstrappingu slovenského dělení ze slovníku 822 878 slov

úroveň	dobře	špatně	chybí	# vzorů	velikost
1	99.24 %	17.17 %	0.76 %	2192	
2	98.08 %	1.52 %	1.92 %	3240	
3	100.00 %	1.16 %	0.00 %	3229	
4	99.94 %	0.01 %	0.06 %	2347	56 kB

Výsledky jedné z iterací generování vzorů jsou v tabulce 1.

Po několika iteracích lze provést závěrečné generování. Vhodnými parametry pro PATGEN lze vygenerovat prostorově úsporné vzory za cenu nižšího pokrytí, nebo naopak maximalistické vzory s nulovou chybovostí a stoprocentním pokrytím.

Parametry generování pro prostorově či výkonnostně optimální vzory nelze v „rozumném“ čase spočítat [16]. Optimu se však dá přiblížit vhodnou heuristikou. Vzory pro dělení americké angličtiny – soubor `hyphen.tex` v každé distribuci TEXu – je daleko od obou optim. Množství výjimek dělení slov k těmto vzorům rychle roste [4,5,6,7,8] a při tomto tempu růstu by zabraly při otištění

[1] Bohužel výslednou množinu slov nelze volně šířit. Volně přístupný seznam slov by umožnil ještě mnohem flexibilnější vytváření variant dělicích vzorů optimalizovaných pro konkrétní projekty.

ještě v tomto století jedno celé číslo časopisu TUGBOAT. Patrně z důvodu zpětné kompatibility nejsou tyto vzory nahrazeny kvalitnějšími, byť kompatibilita je při přidání výjimek do vzorů ve formátu stejně porušena. Dnešní výpočetní technika již umožňuje četné experimenty a generování opakovat s různými parametry. Vhodnými heuristikami nastavení prahů akceptace adeptů vzorů v jednotlivých úrovních generování se lze dostat na mnohem kvalitativně vyšší parametry vzorů, než které docílil před téměř čtvrtstoletím Liang. Typicky je možné za cenu mírného zvýšení velikosti vzorů docílit stoprocentního pokrytí učící množiny, nebo naopak při zadání velikostních omezení na velikost vzorů lze maximalizovat pokrytí. A to vše s nulovou chybovostí a stejnými *konstantními* výpočetními nároky při aplikaci vzorů. Jinak řečeno, počet instrukcí na nalezení dělicích švů slova je ohraničen shora konstantou, nezávisle na tom, z jak velkého slovníku vzory generujeme.

Další technikou, která se dá při generování vzorů použít, je *stratifikace*. Tato technika spočívá v tom, že se snažíme minimalizovat množinu slov k učení, aniž bychom ale přišli o funkčnost vzorů na výjimkách. Máme-li například slovník generovaný morfologickým analyzátorem, tedy známe od každého slovního tvaru slovní základ, stačí do slovníku slov zahrnout náhodně pouze pár slovních tvarů od jednoho lemmatu. Dělení koncovek se zgeneralizuje, neboť koncovkové množiny se neustále opakují a učící algoritmus bude mít dostatek učících příkladů, aby se pravidelnosti dělení konců slov naučil. Naopak se nesmí v seznamu učících slov zapomenout na negace a předpony. Dělení za první slabikou slov začínajících na *na- naj-*, *pre- pred-* apod. je nutno nahlížet jako na výjimky.

5 Shrnutí: čas pro změnu?

Bylo vytvořeno několik variant nových vzorů dělení pro slovenštinu. Vzory jsou pro testování k dispozici ve FTP archívu CSTUGu v adresáři `cstug/sojka/skhyp`. Po nezbytné fázi testování předpokládáme jejich zařazení do běžných TEXových distribucí a projektu OPENOFFICE.ORG a budou šířeny bez omezujících licenčních podmínek.

Jelikož změna vzorů dělení pravděpodobně způsobí změnu zalomení již vytvořených dokumentů, je třeba být v případě rozšířeného požadavku na zpětnou kompatibilitu obezřetný. Jelikož na zálohování *úplných* zdrojů včetně zdrojů potřebných na generování formátu se obvykle zapomíná, při požadavku zpětné kompatibility je třeba zvážit všechna pro i proti a nové vzory si třeba zavést jako nový jazyk (`\language`) spolu se starými. Jsme přesvědčení, že čas pro změnu po více než dekádě používání současných vzorů nastal a kvalita nových vzorů je dostatečným argumentem pro zavedení změny. Po té již ostatně několik let volají také uživatelé OPENOFFICE.ORG a dalších sázecích systémů, kteří dosud používají staré vzory dělení.

Reference

1. David Antoš a Petr Sojka. Generování vzorů dělení slov v UNICODE. V Kasprzak a Sojka [12], strany 23–32.
2. David Antoš a Petr Sojka. Pattern Generation Revisited. V Pepping [19], strany 7–17.
3. David Antoš a Petr Sojka. Generování vzorů pomocí knihovny PATLIB a programu OPATGEN. *Zpravodaj ČSTUG*, 12(1):3–12, 2002.
4. Barbara Beeton. Hyphenation exception log. *TUGboat*, 5(1):15, květen 1984.
5. Barbara Beeton. Hyphenation exception log. *TUGboat*, 6(3):121, listopad 1985.
6. Barbara Beeton. Hyphenation exception log. *TUGboat*, 7(3):146–147, říjen 1986.
7. Barbara Beeton. Hyphenation exception log. *TUGboat*, 10(3):336–341, listopad 1989.
8. Barbara Beeton. Hyphenation exception log. *TUGboat*, 13(4):452–457, prosinec 1992.
9. Pat Hall a Durgesh D Rao, editoři. *Proceedings of EACL 2003 Workshop on Computational Linguistics for South Asian Languages – Expanding Synergies with Europe*, duben 2003.
10. Jiří Haller. *Jak se dělí slova*. Státní pedagogické nakladatelství Praha, 1956.
11. Yannis Haralambous. A Small Tutorial on the Multilingual Features of PATGEN2. dostupné na CTAN jako `info/patgen2.tutorial`, leden 1994.
12. Jan Kasprzak a Petr Sojka, editoři. *SLT 2001*, Brno, Czech Republic, únor 2001. Konvoj.
13. Donald E. Knuth. *The TEXbook*, volume A of *Computers and Typesetting*. Addison-Wesley, Reading, MA, USA, 1986.
14. Jana Chlebíková. Ako rozdeliť (slovo) Československo. *Zpravodaj ČSTUG*, 1(4):10–13, 1991.
15. Ladislav Lhotka. České dělení pro TEX. *Zpravodaj ČSTUG*, 1(4):10–13, 1991.
16. Franklin M. Liang. *Word Hy-phen-a-tion by Com-put-er*. PhD thesis, Department of Computer Science, Stanford University, USA, srpen 1983.
17. Franklin M. Liang a Peter Breitenlohner. PATtern GENeration program for the TEX82 hyphenator. dokumentace programu `PATGEN` verze 2.3 z distribuce web2c na CTAN, 1999.
18. Ján Lieskovský. Systém pro práci se seznamy slov. Bakalářská práce, Masarykova univerzita v Brně, Fakulta informatiky, 2003.
19. Simon Pepping, editor. *EuroTEX 2001*, Kerkrade, The Netherlands, září 2001. NTG.
20. Petr Sojka. Notes on Compound Word Hyphenation in TEX. *TUGboat*, 16(3):290–297, 1995.
21. Petr Sojka. Hyphenation on Demand. *TUGboat*, 20(3):241–247, 1999.
22. Petr Sojka. Competing Patterns for Language Engineering. V Sojka et al. [24], strany 157–162.
23. Petr Sojka a David Antoš. Context Sensitive Pattern Based Segmentation: A Thai Challenge. V Hall a Rao [9].
24. Petr Sojka, Ivan Kopeček, a Karel Pala, editoři. *Proceedings of the Third International Workshop on Text, Speech and Dialogue—TSD 2000*, Lecture Notes in Artificial Intelligence LNCS/LNAI 1902, Brno, září 2000. Springer-Verlag.
25. Petr Sojka a Pavel Ševeček. Hyphenation in TEX – Quo Vadis? *TUGboat*, 16(3):280–289, 1995.